162C51
2016/3/3

Androidアプリケーションにおける GUIユーザビリティ自動評価に向けた属性抽出手法の提案

扇田 昌紀[†1,a)]　小枝 正直[†1,b)]

概要：Android アプリケーションの GUI 構造を解析し，ユーザビリティに影響する属性をクラス分類を用いて統計的に抽出する仕組みを実現した．ユーザビリティの優れたアプリケーションから GUI の特徴および改善手法を学習する事で，ガイドラインのみに依存せず，アプリの特性に応じた改善手法を提案するシステムを目指す．

キーワード：Android，GUI，ユーザビリティ，ユーザインタフェース

Extraction Method of Attributes for GUI Usability Automatic Evaluation of Android Applications

Masanori Ohgita[†1,a)]　Masanao Koeda[†1,b)]

Abstract: Today, a huge number of applications with graphical-user-interface (GUI) is available on various smartphones. However, it is difficult to design a user-friendly GUI for everyone. We aim to provide appropriate improvement tips for the developers of GUI application on smartphones. The GUI usability vary according to the characteristics of each applications. This study describes the method to extract a hidden design model in excellent GUI applications by using a machine learning.

Keywords: Android，GUI，Usability，User Interface

1. はじめに

視覚的ユーザインタフェース (GUI) にユニバーサルデザインが必要とされている．一方，アプリケーション (アプリ) の開発や公開の難易度は低下傾向にあるが，UI デザインの素養を持たない開発初心者は使いやすい GUI の設計が容易でない．従来，GUI の評価手法として，アンケートを基とした手法 [1] や開発者自身によるテストが採られてきた．これらの手法は開発初心者には採用が困難である．

本研究ではユーザビリティの優れた GUI をモデル化しておき，自動的にユーザビリティを評価し改善手法を提示可能なシステムを提案する．このシステムでは一般的な GUI デザインガイドラインから得られない，優れた GUI アプリに共通する隠れた使いやすさを機械学習で抽出する．この抽出結果を基とし，アプリのユーザ層などの特性に応じた改善手法を提示するシステムを目指す．

2. システムの実現に向けて

本研究ではデザイン全般あるいはユーザインタフェースデザインにおける概念を応用した．一つ目は “デザインの四原則” というデザイン全般における重要な概念である．この概念は近接，反復，コントラスト，整列の 4 要素 [2] から成り，GUI でも多用されている．Fig.1 に示す GUI の例では，ユーザが選択した画像とそれ以外の画像を区別するために，周囲の背景色を変えてコントラストを与えている．しかし，表現手法は必ずしも如何なる GUI においても最適とは限らない．またコントラストを与える手法は一通りで

[†1] 現在，大阪電気通信大学
Presently with Osaka Electro-Communication University
[a)] mt15a001@oecu.jp
[b)] koeda@isc.osakac.ac.jp

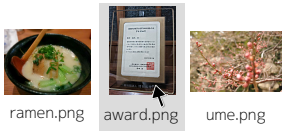

Fig. 1 An Example of The Four Principles of Design in GUI

はない．他にも要素のサイズや間隔を相対的に大きくするなど無数に考えられる．アプリのデザイン性を活かすにも臨機応変なユーザビリティ改善手法が必要不可欠である．

もう一つの概念は “GUI デザインガイドライン” である．主要なプラットフォームにはアプリの GUI デザインを行う上での指針が存在する．スマートデバイス向けプラットフォームにおいては，Android では “Android Design”[3]，iOS では “iOS Human Interface Guidelines”[4] が挙げられる．ガイドラインから定義を作成すれば，ユーザビリティを機械的に評価可能だが，定義のみでは評価が画一的になりかねない．

我々は ROBOMECH 2015 [5] にて，ガイドラインに依存しないユーザビリティ評価手法を提案した．この過去提案では実現に向けた第一段階として，“静的解析” という独自の GUI 解析手法も実装した．更に検証実験としては，複数の Android アプリに対して解析および特徴抽出を行った．しかし，解析が高速である反面，独自である為に解析可能な GUI 部品 (View) に制約があり，解析不可能なアプリも散見された．そこで本報告では，解析手法を次章のように刷新し，新たなシステムを実現した．

GUI のユーザビリティ自動評価および改善手法の提示を実現するために，大きく以下のステップに分けて研究を進めている．

(1) GUI の属性および改善手法を抽出
　(a) 属性解析　(属性の抽出)
　(b) 動作解析　(改善手法の抽出)
(2) 属性のクラス分類 (機械学習による GUI の特徴抽出)
(3) 特徴および改善手法をクラスタリング

本報告で刷新した項目は，このうち 1-(a) および 2 である．今回の刷新によって，理論上全ての View および 全てのアプリが解析可能となった．また比較可能な属性数が 5 件から 27 件へと増加した．

3. GUI の属性解析

本研究における GUI の属性解析では，アプリの GUI 構造を独自のツリー形式へ変換し，各 GUI パーツの属性や親子関係, 反復関係などを調査する．

初めに解析対象とするアプリの実行環境として Android を採用した．理由の一つは Android 自体がオープンソースであり，仕様の大部分が公開されているためである．次にアプリの配布方法に制限がない為，数も多く品質も多種多様であり，ユーザビリティの改善効果が見込める可能性も高い．従って，本稿では特に Android アプリに対する属性解析について述べる．

3.1 アプリの解析 および GUI ツリーの抽出

初めに 解析対象の Android アプリの配布ファイル (APK ファイル) を用意し，“Android Virtual Device” (AVD) と呼ばれる仮想マシン上でファイルをインストールする．次に “Hierarchy Viewer” というツールを拡張し，アプリの GUI 構造を解析する独自拡張版を開発した．これにより Code 1 に挙げるような GUI ツリー および， 各 View から Fig. 2 のようなキャプチャ画像を取得する．

Hierachy Viewer のオリジナル版は，“Android SDK” および 総合開発環境 (IDE) 用プラグイン “Android Development Tools” (ADT) に同梱される．IDE 用プラグインに同梱されるという性質上，IDE から呼び出して使用する事を前提とし，手動操作のみに対応している．対して独自拡張版は，本稿執筆者が以下の拡張を施したものとなっている．

- スタンドアロン化 — Android SDK や ADT から必要最低限のプロジェクトを抽出してアーカイブ化し，ビルドを簡略化
- コマンドラインオプションによる自動制御の対応
- GUI ツリーの JSON 形式によるダンプ
- 全ての View 毎の PNG 形式によるキャプチャ

3.2 GUI ツリーからの属性抽出

前節で生成した GUI ツリーから属性を抽出する為，各々の View より属性値を計算する．GUI ツリーから各々の View を取り出し，データセット生成器である以下の Perl モジュールで処理を行う．全ての View に対してこれを繰り返すと，最後に各モジュールより当該画面内のデータセットが得られる．尚，これらの Perl モジュールは全て本研究で実装したものである．

- 汎用データセット生成器 — GUIRefine::DataSetGenerator::Android::General
- View 種別毎のデータセット生成器 — 例: GUI::DataSetGenerator::Android::Button (Button 系の View を処理する場合)

以上のうち，前者の汎用データセット生成器が算出する属性の一部を次に挙げる．

- numOfColors — 画面内の色数
- most1stColor — 画面内で最も使われている色
  但し，主要な 12 色のリストより選択された近似色の色名となる．
- most2stColor — 画面内で 2 番目に使われている色

次に後者の Button 用データセット生成器が算出する属性の一部を以下に挙げる．

- num — 画面内のボタンの数

- minWidth — 画面内の最も小さなボタンの幅
- minHeight — 画面内の最も小さなボタンの高さ
- minDistTop — 画面上端からボタンまでの距離
- minDistLeft — 画面左端からボタンまでの距離

4. 実験 - 複数アプリの属性解析＆クラス分類

属性解析による特徴抽出を実現する為に実験を行った．具体的には，属性解析を複数のアプリに対して行い，算出されたデータセットから属性値の傾向を R 言語を通し Random forest アルゴリズムにより比較した．実験対象のアプリはカテゴリを絞り，Google Play ストア上で無料配布されている電卓系アプリであり，進数計算等の特殊な電卓は除外した．アプリの一覧を Table 1 に，比較対象とする GUI 属性を Table 2 に挙げる．名称や作成者，その他の欄は，2015 年 12 月現在の Google Play ストア 日本語版または英語版における情報を記載した．所属欄は本実験における分類変数であり，A が使いやすいアプリ，B が使いにくいアプリを示す．この使いやすい，使いづらいという条件は評価スコアやレビュー文章を参考に人為的に設定した．例えば，“使いやすい” という文章があれば好評，“使いづらい” や “使い方がわからない” 等は悪評した．あくまでも本実験を行う為の一つの設定値である為，当該アプリや関係者を奨励あるいは中傷する意図はない．また，本実験では対象とする画面をアプリ 1 本につき 1 つとし，主には起動時の画面に絞る．

本実験結果のグラフの一部を Fig.3， Fig.4 に示す．まず Fig.3 は分類変数からクラス分類した結果，どの GUI 属性が高い関係性を持つ可能性があるかを示すグラフである．単一の図に 2 つのグラフが含まれているが，左側 ($MeanDecreaseAccuracy$) は特徴量加工による重要度，右側 ($MeanDecreaseGini$) はジニ係数による重要度を意味する．参考として Python の機械学習ライブラリ “scikit-learn” ではジニ係数による重要度のみが使用される．次に Fig.4 は，全属性中 8 種類の分類変数別の頻度分布である．

過去研究 [5] ではアプリのカテゴリを限定せず，正しく解析可能なアプリに絞って実験を行った．結果，ボタンの高さが大きいアプリは概ね A に所属する事が読み取れ，デザインガイドラインで推奨されるサイズに一致した．対して今回はカテゴリを絞った為か，現時点の 27 種類の属性からは明確な視点が得られていない．しかし Fig.4 からも判るように各属性ともに正しい値が得られている．また選定した全てのアプリに対して解析が行えた事は本報告において成功といえるであろう．今回の結果を踏まえ，更に比較属性数を増やすとともに，分類変数を設定するためのより人為的でない手法の実現に取り組むべきであると考える．

5. おわりに

本研究ではガイドラインのみに依存せず，機械学習により自動的にユーザビリティを評価し改善手法を提示可能なシステムを提案した．第 2 章に挙げた手順のうち，GUI の属性解析および特徴抽出を実現している．具体的には独自拡張版 Hierarchy Viewer で GUI ツリーを抽出し，GUI 評価に必要な属性の算出を可能にした．さらに属性解析を複数のアプリに対して実行し，Random forest でクラス分類を行う事で，統計処理する実験も行った．

第 2 章 のとおり，我々の過去研究では，GUI を解析する為にレイアウトを独自解析してきたが，精度上の問題があった．今回実現した手法では Android SDK に含まれるツールを拡張する事で精度を改善した．またアプリの逆コンパイルを行うことなく，ライセンスに配慮可能な点も特徴といえる．将来，多くのアプリが各々のユーザ層にとって更に使いやすくなるとともに，開発者の負担軽減に繋がることを願う．

Code. 1 GUI Tree which generated by Customized Hierarchy Viewer


```
{
"zero": {
        "image": "/tmp/gui-refine-cache/latele.nekocalc/zero.png",
        "name": "android.widget.Button",
        "property_drawing:getAlpha()": "1.0",
        "property_drawing:getRotation()": "0.0",
        "property_drawing:getSolidColor()": "0",
        "property_drawing:getX()": "0.0",
        "property_drawing:getY()": "99.0",
        "property_getVisibility()": "VISIBLE",
        "property_isActivated()": "false",
        "property_isClickable()": "true",
        "property_isEnabled()": "true",
        "property_isHovered()": "false",
        "property_isSelected()": "false",
        "property_isSoundEffectsEnabled()": "true",
        "property_layout:getBaseline()": "69",
        "property_layout:getHeight()": "99",
        "property_layout:getWidth()": "118",
        "property_layout:hasTransientState()": "false",
        "property_layout:isLayoutRtl()": "false",
        "property_layout:layout_bottomMargin": "0",
        "property_layout:layout_gravity": "NONE",
        "property_layout:layout_height": "0",
        "property_layout:layout_leftMargin": "0",
        "property_layout:layout_rightMargin": "0",
        "property_layout:layout_topMargin": "0",
        "property_layout:layout_weight": "50.0",
        "property_layout:layout_width": "MATCH_PARENT",
        "property_layout:mBottom": "198",
        "property_layout:mLeft": "0",
        "property_layout:mRight": "118",
        "property_layout:mTop": "99",
        "property_measurement:mMinHeight": "96",
        "property_measurement:mMinWidth": "128",
        "property_padding:mPaddingBottom": "0",
        "property_padding:mPaddingLeft": "0",
        "property_padding:mPaddingRight": "0",
        "property_padding:mPaddingTop": "0",
        "property_scrolling:mScrollX": "524229",
        "property_scrolling:mScrollY": "0",
        "property_text:getTextSize()": "52.0",
        ....
    },
    ....
}
```


Fig. 2 Image of view which captured by Customized Hierarchy Viewer

参考文献

[1] 仲川薫，須田亨，善方日出夫，松本啓太，“ウェブサイトユーザビリティアンケート評価手法の開発”，pp.1-2，ヒューマンインタフェース学会，2000.
[2] 中川總，“グラフィックデザイナーのためのユニバーサルデザイン実践テクニック 51”，ワークスコーポレーショ

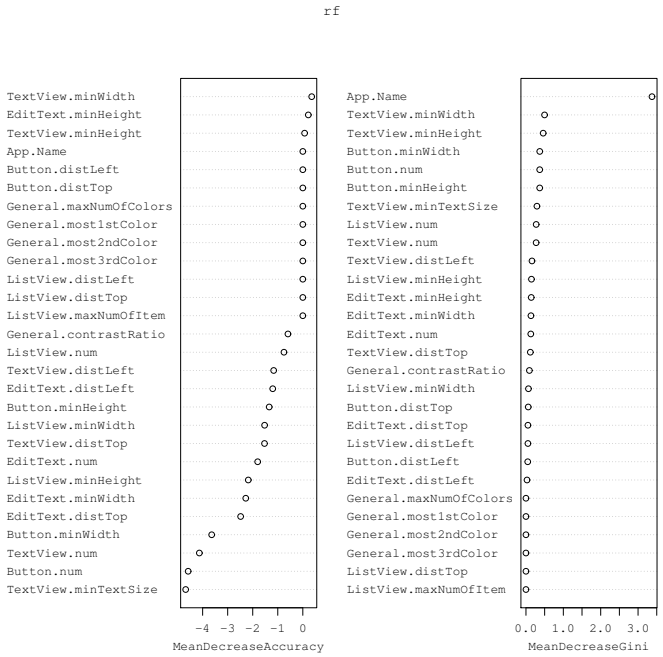

Fig. 3 Experiment Result - Important Attributes

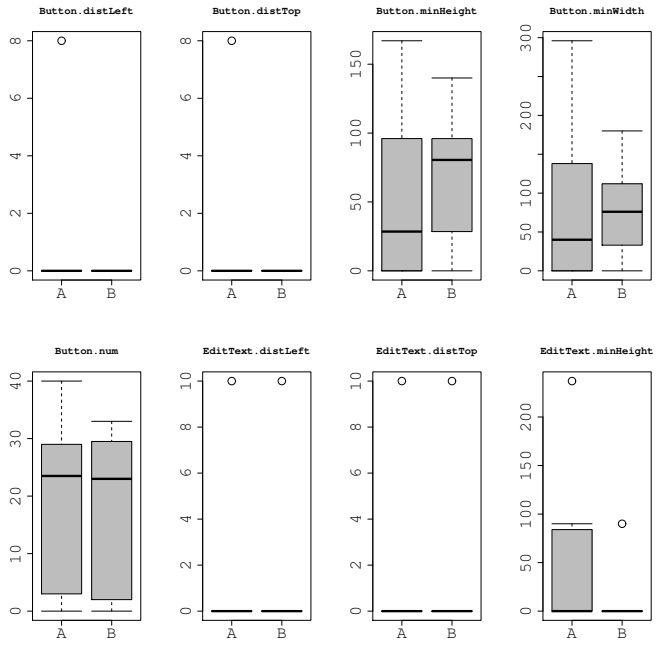

Fig. 4 Experiment Result - Frequency Distribution (Excerpted)

Table. 1 Experimented Apps

名称	作成者	バージョン	評価スコア	所属	UI好評	UI悪評
Calculator	andanapps	2.0.5	4.1	A	-	-
Calculator Pro free	Apalon Apps	2.0	4.1	A	2	5
CALCU Free	Designer Calculators	2.1.0	4.5	A	-	-
Calculator Plus Free	Digitalchemy, LLC	4.9.2	4.4	A	23	2
Simple Calculator	Kugelschreiber	3.0	4.1	A	-	-
LateCalc - Calculator	Latele Inc.	1.5.0	3.6	B	5	3
NekoCalc - Calculator	Latele Inc.	1.3.0	4.3	A	1	-
鷹の爪電卓	Loop-Sessions.LLC.	1.31.0	3.8	B	-	-
Colorful calculator	n225.zero	1.6	3.9	A	23	-
Panecal Scientific Calculator	Appsys	6.0.2.2	4.1	A	3	1
TigerCalc	tripot	1.1	3.4	B	-	-
RealCalc Scientific Calculator	Quartic Software	2.3.1	4.5	A	-	-
超シンプル電卓 2	Nobuo _ CREATE	1.3	3.9	B	-	-
Pretty Calculator	Pretty Pet Co	1.1.3v	4.0	A	-	-
お父さん電卓	SoftBank Corp.	3.0.0	3.6	B	1	3
Calculator	7th Gear	1.0	3.9	B	-	-
Calculator	TricolorCat	1.10.7	4.1	A	8	2
Calculator	woodsmall inc.	1.6.5	4.0	A	13	3

Table. 2 Comparison Attributes (Excerpted)

項目名	説明	単位
Button.num	一画面におけるボタンの数	個
Button.distTop	画面上端からの距離 (最低値)	px
Button.dLeft	画面上端からの距離 (最低値)	px
Button.minHeight	ボタンの高さ (最低値)	px
Button.minWidth	ボタンの幅 (最低値)	px

ン，2011.
[3] Google, “Android Design”, 2014. https://developer.android.com/intl/en/design/index.html (最終閲覧日: 2015/12/16)
[4] Apple, “iOS Human Interface Guidelines”, 2014. https://developer.apple.com/jp/devcenter/ios/library/documentation/UserExperience/Conceptual/MobileHIG/BasicsPart/BasicsPart.html (最終閲覧日: 2015/12/16)
[5] 扇田昌紀，小枝正直，“機械学習によるスマートフォンにおける GUI ユーザビリティの自動評価”，日本機械学会ロボティクス・メカトロニクス講演会 2015 (ROBOMECH 2015) ，2015.